

**Module Name: Microprocessor Systems Laboratory**

**Module Code: ELCE333**

**Laboratory Experiment No. 1**

**Expriment Tite: Microcontroller Assembly Program Development**

**Pre-lab Report**

**Group Members**

Name: Afra Bin Fares ,ID#:100033139

Name: Alya Humaid AlAlili ,ID#100037087

Name:Anoud Alshamsi ,ID#100035514

**Instructors**

Mohammed Ali Saif Al Zaabi

Mahmoud Khonji

**Spring 2015**

1. **Introduction:**

Assembly language is a language that allows users and microcontrollers to converse between each other and it’s the basic language for processors. The user deals with instructions that are based on the physical CPU. CodeWarrior IDE is used in this lab, which is a software tool that sets up the connection with the micro controller and run the assembly language accordingly. The communication needs several elements to be present in order for it to be successful and error free. Those elements are:

**a. Assembly language syntax**

Those are the regulations that state how expressions are formed into statements that consist of the assembly language.

**b. Labels**

A text that represent the address in ROM or RAM

**c. Instruction**

Defined by the manufacturer of the micro controller which means the programmer needs to learn the rules of the micro controller he's using. The instructions performed in this lab are arithmetic and data transfer instructions.

**d. Operands**

It is the argument or value on which the instruction performs at like a variable, register, label or memory address.

**Aim:** The main aim of this pre lab experiment is to be familiar with the process of editing, assembling, simulating, downloading and executing an assembly program.

**Objectives: for this pre lab experiment it is expected to:**

1. Read the dragon12\_plus\_usb\_9s12\_manual

2. Execute the steps described in Relocatable Assembly Tutorial to learn how to use CodeWarrior for the development of Assembly code .

3. Using the assembly shell program , simulate the given program.

4. Examine the contents of the memory locations 0x1000 and 0x1010 and the accumulators before and after stepping in each line of the program and record them.

5. Answer the pre-lab Questions

**Risk Assessment:** Low

**2. Pre-Lab Tasks:**

In this pre-lab experiment the following steps were followed:

1) Using the assembly shell program, the following program was copied into the main.asm file, compiled and simulated:

; Include derivative-specific definitions

INCLUDE 'derivative.inc'

; export symbols

XDEF Entry

; Insert here your data definition.

; code section

Entry:

LDAA #$FF; Initialize Accumulator A with 0xFF

STAA $1000; Store the value 0xFF in memory location 1000

CLRA; Initialize Accumulator A with 0x00

STAA $1010; Store the value 0x00 in memory location 1010

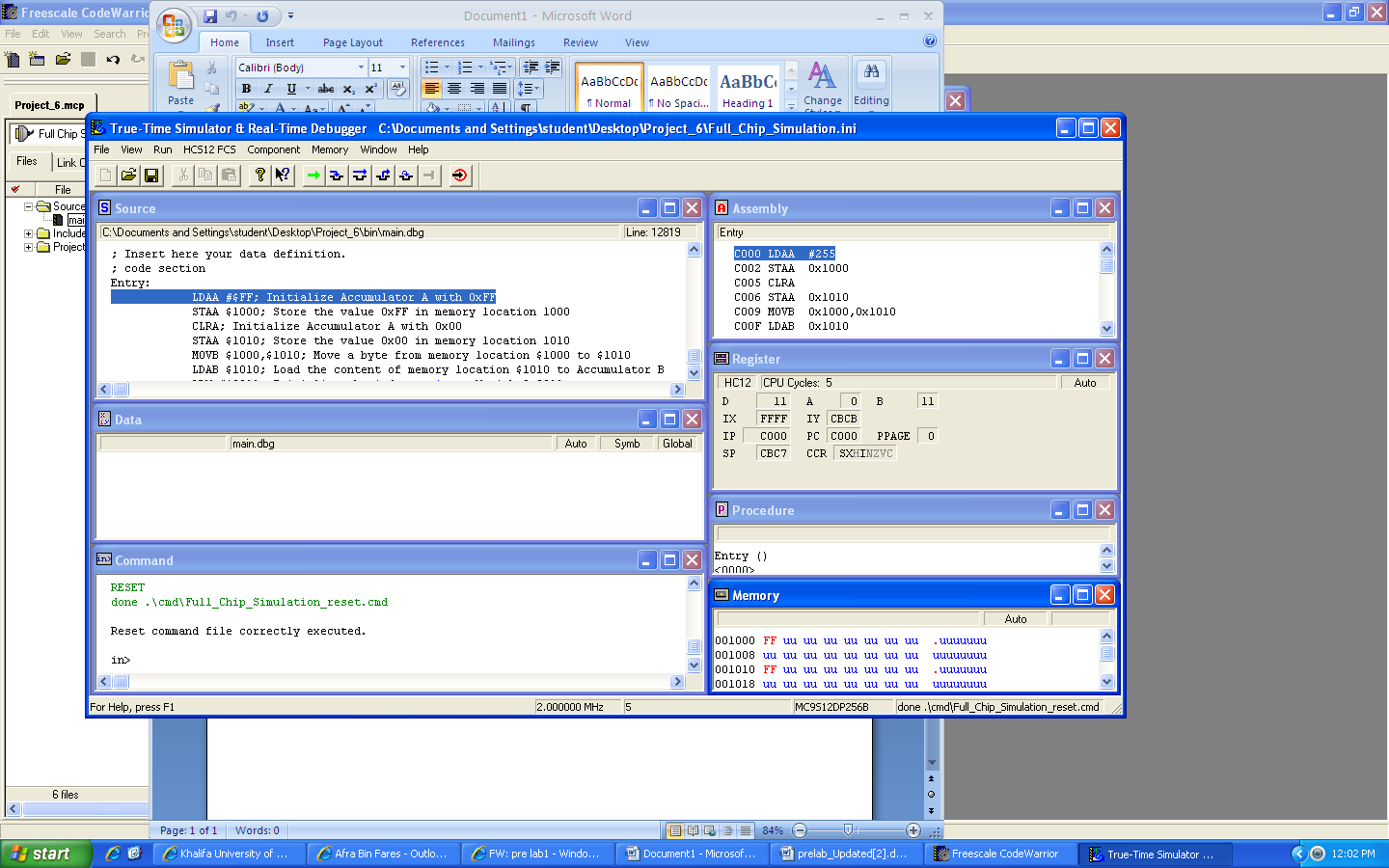
MOVB $1000,$1010; Move a byte from memory location $1000 to $1010

LDAB $1010; Load the content of memory location $1010 to Accumulator B

LDX #$0011; Initialize the index register X with 0x0011

EXG X,B; Exchange the contents of Accumulator B and index register X

2) After simulating the given program the contents of the memory locations 0x1000 and 0x1010 and the accumulators (before and after stepping in each line of the program) were examined (the next figure shows where each value was observed at).



Program

Accumulator A

Accumulator B

Index register

Memory location 1000

Memory location 1010

Note: the results outputted in the window represent the values of the previously highlighted line.

The following table shows the results observed as the program was step by step executed:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table 1 Program Step by Step Execution** | | | | | |
| Instruction | Memory Location 0x1000 | Memory Location 0x1010 | Accumulator A | Accumulator B | Index Register X |
| LDAA #$FF | FF |  | FF |  |  |
| STAA $1000 | FF |  | FF |  |  |
| CLRA | FF | FF | 0 |  |  |
| STAA $1010 | FF | 00 | 0 |  |  |
| MOVB$1000,$1010 | FF | FF | 0 | 11 |  |
| LDAB $1010 | FF | FF | 0 | FF |  |
| LDX #$0011 | FF | FF | 0 | FF | 11 |
| EXG X,B | FF | FF | 0 | 11 | FFFF |

3) The following instructions of the program operated on the mentioned addressing mode:

a) LDAA #$FF Immediate Addressing mode , where the content of memory location at $FF immediately gets added to(load it to) accumulator A.

b) CLRA Inherent Addressing mode, where the content of A gets cleared out.

c) LDAB $1000 Extended Addressing mode, where it fetches the content of memory location at $1000 and puts it (load it) in accumulator B.

**3. Pre-Lab Questions:**

1.

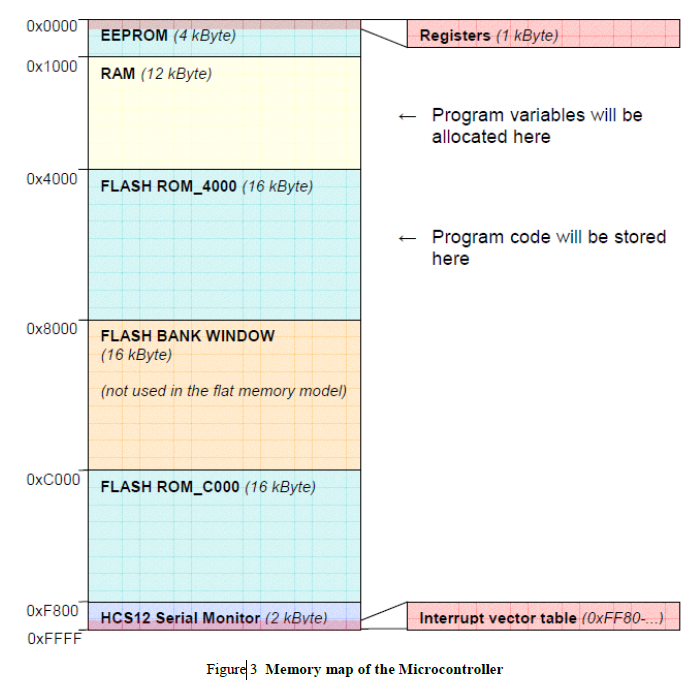
a. How much RAM does your microcontroller have? 12 K Byte

b. How much EEPROM ? 4 K Byte

c. How much Flash? 256 K Byte

d. Does it have any other kind of memory? Yes , registers ….

2. Draw the memory map of your microcontroller.



3. What registers does your microcontroller have?

* Index register X , Index register Y , Stack pointer ,program counter , condition code register , accumulator A , accumulator B , double accumulator D,Device ID register, FLASH control register, EEPROM control register, and other special function registers located at the first 1024 bytes of the memory( from $0000 to $03FF).